T.C.

MARMARA UNIVERSITY FACULTY OF ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

GROUP 10

Beyzanur Çabuk 150119632

Bihter Nilüfer Akdem 150119810

Eşref Emre Koca 150119798

Safa Şan 150119685

Senanur Yılmaz 150119801

Sena Nur Boylan 150119737

Şule Koca 150119057

Zeynep Destan 139019026

Supervised By:

Prof. Murat Can Ganiz

TABLE OF CONTENTS

1. Introduction

The purpose of the application we develop will be to create the University registration system simulation via Python.

1.1 Purpose of the System

Purpose of the application It brings the course registration process approval processes to the system quickly and smoothly.

1.2 Scope of the System

The purpose of the course record analysis system simulation; It is to create functions such as creating course registration, grading, creating transcripts in an easy and effective way. This project is based on Marmara Computer Engineering Department courses and the prerequisite tree is determined accordingly. Progressed in accordance with OOP Standards

1.3 Objectives and Success Criteria of the Project

The primary objectives of our program. Adhering to Marmara University Engineering faculty prerequisites and course selection procedures. Developed based on development and clean code in line with OOP standards.

1.4 SRS Standards

The Requirements Analysis Document will confirm to the IEEE standard IEEE Std 830-1998.

IEEE Computer Society. (1998, October 20). IEEE Recommended Practice for Software Requirements Specifications. IEEE Std 830-1998. IEEE Xplore.

Krüger, C. L. (2021, December 16). How to Write a Software Requirements Specificiation (SRS Document). Retrieved from www.perforce.com: https://www.perforce.com/blog/alm/how-write- software-requirementsspecification-srs-document

1.5 Overview

A student registration system simulation developed over Python based on the BYS system. Our project has a single user type experience. Based on the student model, student course selection and registration processes are available in our application.

2. Proposed System

The project, which proceeds on the basis of the BYS system simulation, embraces minimal values. Developed in accordance with the OOP system and clean code. It is aimed to create a student registration system simulation in the most optimum and simple way.

2.1 Functional Requirements

This project is a small prototype of course registration analysis system simulation.

The modelling will contain the followings:

The database contains students' names, surnames, departments and 'student ID' information.

In the Course class, the names of the courses, course codes, credits and prerequisites are stored.

Student:

StudentID, First Name, Last Name, Semester.

Student informations like StudentID, isApprove, ScheduleStatus etc. kept in separate .JSON files

Course:

Course Code, Course Name, Credit, Prerequisite, Course Type.

Advisor:

Advisor ID, Advisor Name, Advisor Password, Advisor E-mail.

- To approve registration for enrollment

Transcript:

Transcript_ID, GPA, Total Credit, Completed Credit , Semester, Student_ID

- Getting GPA
- Adding semester
- Failed courses
- Passed courses

CourseRegistrationSystem:

Course, Grade, Successful

- Setting quota of course after registration
- Setting total credit of student after registration

CourseRegistrationSimulation: It simulates the registration system.

Log: This class logging the errors. It creates a log file after registration and shows why students didn't register the courses.

Person: The class is superclass of Advisor and Student classes.

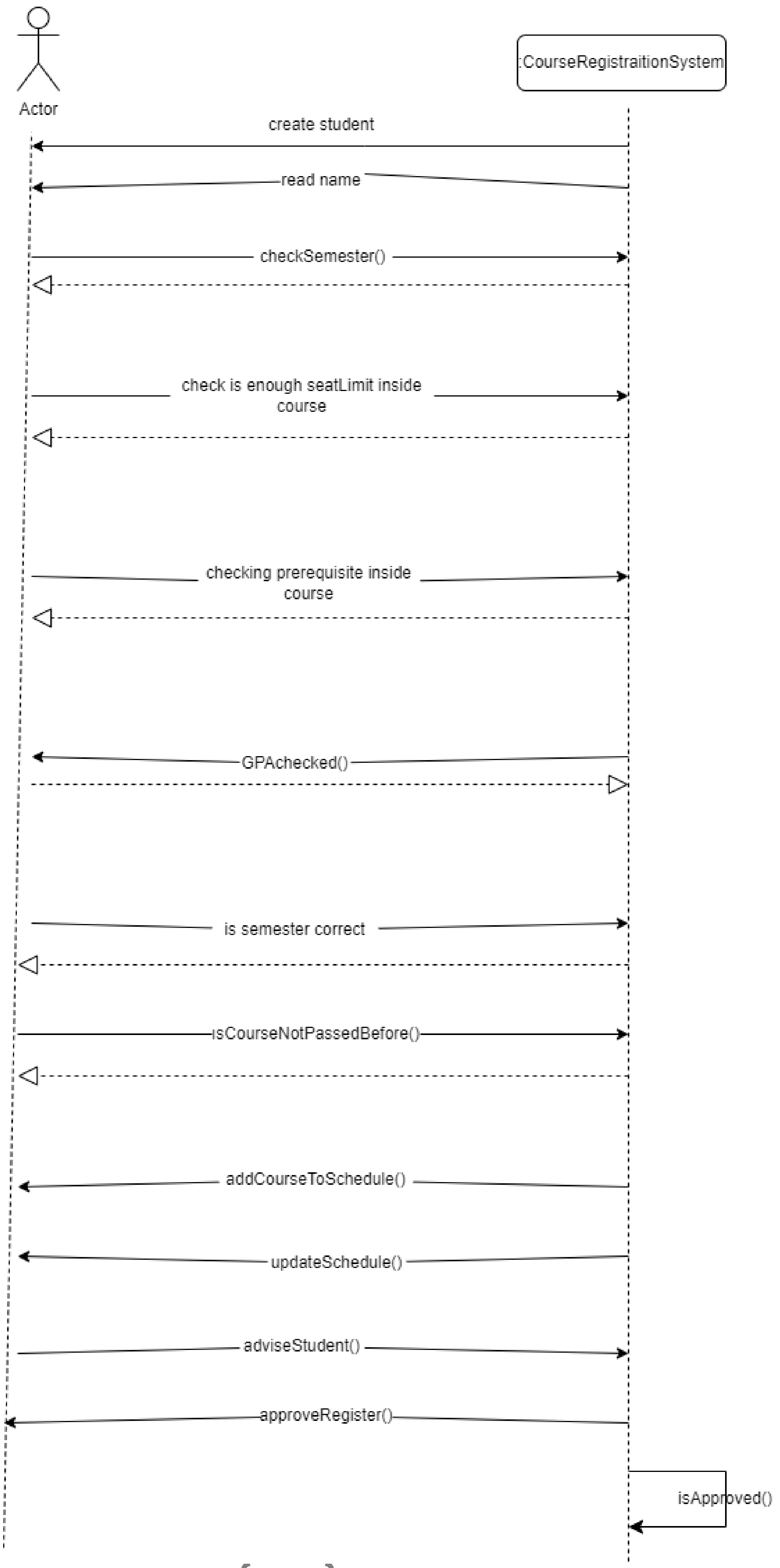
RandomStudentAccSemester: The class allows the program to randomly generate students and transcripts according to semester

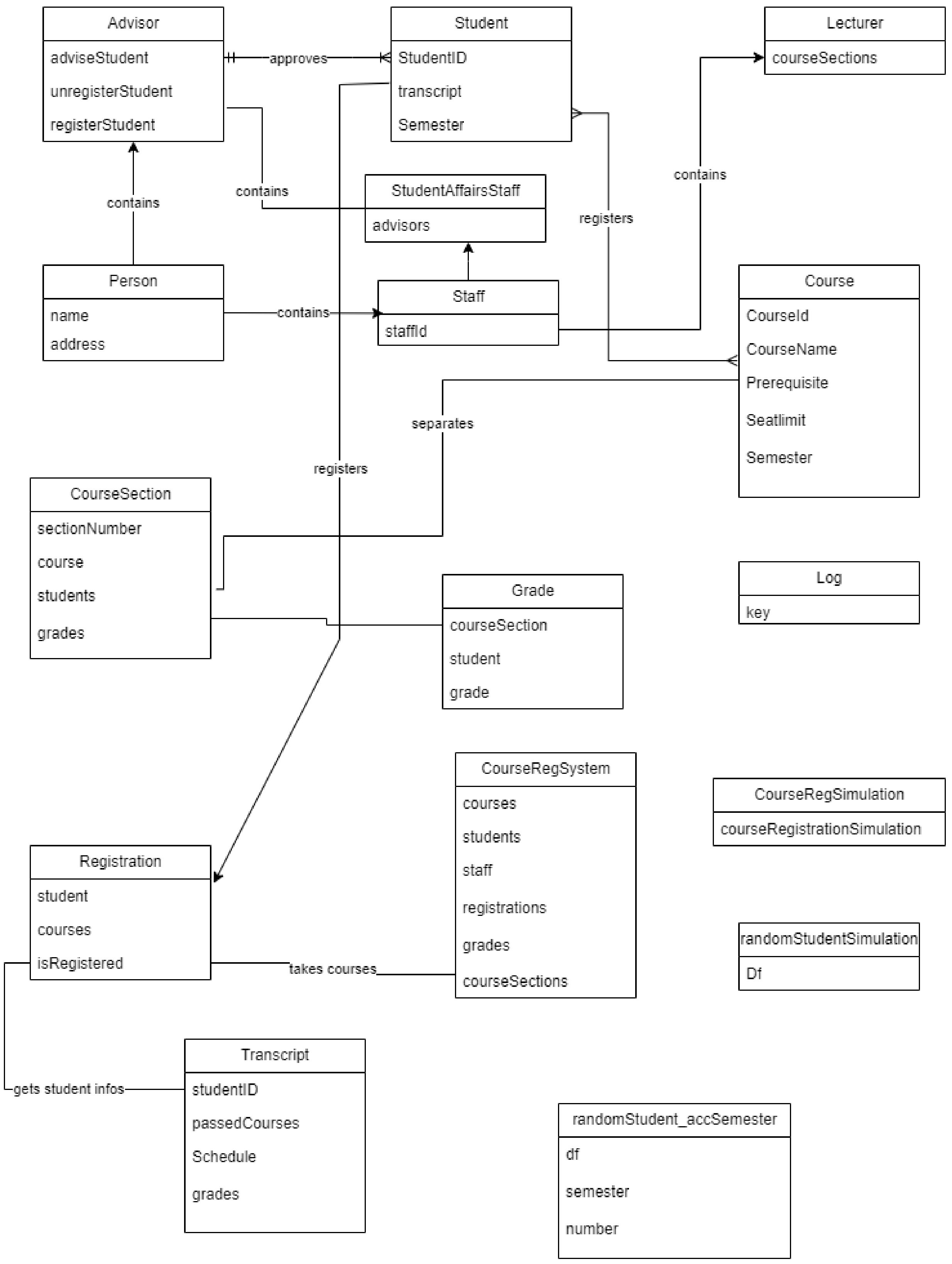
RandomStudentMix: The class allows the program to randomly generate students and transcripts independently semester

Registration: It checks the conditions like semester, prerequisites etc.

Tests: It provides a unit test.

SSD:


Actor
:CourseRegistraitionSystem
create student
read name
checkSemester()
check is enough seatLimit inside course
checking prerequisite inside course
GPAchecked()
is semester correct
isCourseNotPassedBefore()
addCourseToSchedule()
updateSchedule()
adviseStudent()
approveRegister()
isApproved()



Domain Model:

2.1.1Use Case

Use Case: Course Registration

Actors: Student, System, Advisor

Stakeholders and Interests:

Student: Chooses the courses he/she wants to enroll in during the semester

Advisor: Approves the student's chosen courses

Registration System: Establishes the connection between the advisor and the student

Preconditions:
Student’s past transcript information must be generated before registration.
Courses must be available in the registration system.

Main Success Scenario:

1. Student login the system.
2. Choose the course to enroll.
3. The system checks that the course belongs to the correct semester.
4. If it is correct, system checks if there any overlap for this course.
5. If system didn’t find any overlap, system checks that does student. complete the prerequisites of chosen course.
6. If student completed, system checks the course type.
7. If course is mandatory course, system sends the course selection to student’s advisor.
8. If course type is TE, advisor checks the student's credits are enough to take the TE course.
9. If course type is EP, advisor checks the student's credits are enough to take the EP course.
10. If student has enough credit, advisor checks the student for over credit.
11. If student has not over credit, advisor checks quota for selected course.
12. If the course has enough quota, advisor approve course selection.
13. System adds the course to transcript.

Alternative Use Case Scenarios:

1. The student enters the system.
2. The student chooses the course to enroll in from the courses offered to him/her.
3. The system sends the course selection to student's advisor.
4. The advisor approves the student's course.
5. The system adds the course in the transcript.

Alternative Scenario 1: Course Selection Errors
2a. At step 2, the student cannot choose the course of the wrong semester.

Alternative Scenario 2: System Check Errors
3a. At step 3, if there is an overlap for the selected course, the system will not allow the student to take the course.

3b. At step 3, if the selected course has a prerequisite, the system checks whether the student has successfully passed this prerequisite course. The system will not allow the student to submit the course for which he/she has not successfully passed the prerequisite for the approval of the advisor.

Alternative Scenario 3: Advisor Check Errors
4a. At step 4, if the selected course is TE, FTE or Graduation Project course, it is checked whether the student has enough credits to take these courses. The advisor will not approve the course of the student who is not eligible to take the course.

4b. At step 4, if the total credits of the courses chosen by the student are more than the maximum credits, the advisor will not approve the course. 4c. At step 4, if the total quota of the course chosen by the student is full, the advisor will not approve the course.

2.1.2Product Functions

System simulation has the functions:

- View Transcript
- Choose a Course
- View Remaining Courses
- Add Lesson
- Check Prerequisite
- Create log
- View log

2.1.3 Operating Environment

-Visual Studio Code

- Python

- GitHub

-Trello

-Discord

2.1.4 Assumptions and Dependencies

-Not specific database usage. Manually adding .json data.

- Total of years 4, each year has two semesters.

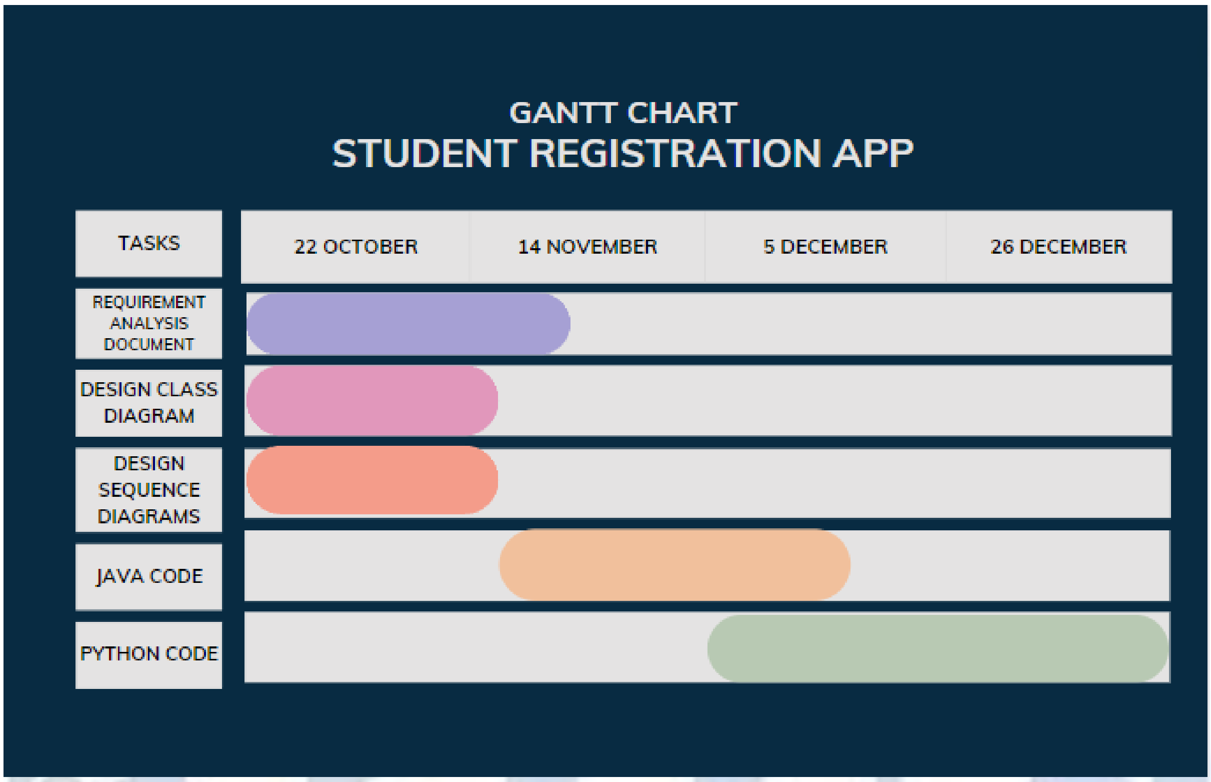
2.1.5 Hardware Interfaces

-Console Application.

2.1.6 Software Interfaces

Software Utility	Definition
Programing Language	Python
IDE	Visual Studio Code
Operating System	Windows
Data Format	.json

3. Project Schedule



4. Glossary

Pre-requisite: A course or other requirement that a student must have taken prior to enrolling in a specific course or program.

Credit: Recognition for having taken a course at school or university, used as measure if enough hours have been made for graduation.

Semester: A calendar that divides the academic year into 15-17 weeks terms. There are generally two semester per academic year: Fall (beginning in August or September) and spring (beginning in January)

Advisor: A person who gives advice in a particular field.

Course: In higher education a course is a unit of teaching that typically lasts one academic term, is led by one or more instructors (teachers or professors), and has a fixed roster of students. A course usually covers an individual subject.

Affair: An event or sequence of events of a specified kind or that has previously been referred to.

Transcript: A transcript is an official document that provides an inventory of courses and grades earned by a student throughout their academic career.

Simulation: Imitation of a situation or process.

Log: In computing, logging is the act of keeping a log of events that occur in a computer system, such as problems, errors or just information on current operations.

Registration: The action or process of registering or of being registered.

Object-oriented: A computer programming model that organizes software design around data, or objects, rather than functions and logic

Superclass: In object-oriented programming, a class from which other classes inherit code is called a superclass

5. References

Medium, “Requirement Analysis Document (RAD) Nedir, Nasıl yazılır?” 8 Oct 2022 https://medium.com/@emre_karaoglu/requirement-analysis-document-rad-nedir-nas%C4%B1l-yaz%C4%B1l%C4%B1r-faf4871986ab.

Florida State University, “Requirements Analysis Document” 8 Oct 2022 https://www.cs.fsu.edu/~lacher/courses/COP3331/rad.html

Wrike, “How to Carry Out a Requirements Analysis” 8 Oct 2022 https://www.wrike.com/blog/how-carry-out-requirements-analysis/#What-is-a-requirements-analysis